Introducción a Ruby on Rails

Arrays

Overview

- Arrays
  - Cómo se crean
  - Cómo se modifican
  - Acceso a elementos

Arrays

- Son colecciones de **referencias a objetos** auto-expandibles.
- Indexadas utilizando el operador **[]**
- Puede ser indexado con **números negativos** o **rangos**
- Se permiten tipos de datos **heterogéneos** en el mismo array.
- Se puede utilizar %w{string1, string2} para la creación de arrays de string.

Arrays

```ruby
array_hetero = [1, "dos", :tres] # datos heterogeneos
puts array_hetero[1] # => dos (los indices empiezan en 0)

arr_palabras = %w{ que lindo dia es hoy! }
puts arr_palabras[-2] # => es
puts "#{arr_palabras.first} - #{arr_palabras.last}" # => que - hoy!
p arr_palabras[-3, 2] # => ["dia", "es"]
#(volver atrás 3 y tomar 2 posiciones)

# (Se explica el tipo de datos range en otra clase...)
p arr_palabras[2..4] # => ["dia", "es", "hoy!"]

# Obtener un String separado por ‘,’ del array
puts arr_palabras.join(',') # => que,lindo,dia,es,hoy!
```

Modificando Arrays

- Modificando Arrays
  - Append (agregar al final): **push** o **<<**
  - Eliminar: **pop** or **shift**
  - Set: **[]=** (método)
- Seleccionar elemntos randomicamente **sample(number_of_elements)**
- Ordenar o revertir el orden de un array con **sort!** y **reverse!**

Arrays, pilas y colas

```ruby
# Comportamiento de pila/stack (LIFO)
stack = []; stack << "uno"; stack.push ("dos")
puts stack.pop # => dos

# Comportamiento de cola/queue(FIFO)
cola = []; cola.push "uno"; cola.push "dos"
puts cola.shift # => uno

a = [5,3,4,2].sort!.reverse!
p a # => [5,4,3,2] (modifica el array)
p a.sample(2) # => obtiene 2 elementos aleatoriamente

a[6] = 33
p a # => [5, 4, 3, 2, nil, nil, 33]
```

Api

- Cuenta con muchos métodos útiles!
  - each – itera a través de un array
  - select - filtra un array con una selección
  - reject - filtra un array con un rechazo
  - map - modifica cada elemento dentro del array

http://ruby-doc.org/core-2.2.0/Array.html

Array

Arrays are ordered, integer-indexed collections of any object.

Array indexing starts at 0, as in C or Java. A negative index is assumed to be relative to the end of the array---that is, an index of -1 indicates the last element of the array, -2 is the next to last element in the array, and so on.

Creating Arrays¶ ↑

A new array can be created by using the literal constructor `[]`. Arrays can contain different types of objects. For example, the array below contains an Integer, a String and a Float:

```
ary = [1, "two", 3.0] #=> [1, "two", 3.0]
```

Procesamiento de arrays

```ruby
a = [1, 3, 4, 7, 8, 10]

# itera sobre el array y ejecuta el bloque
# print imprime sin salto de línea
a.each { |num| print num } # => 1347810
puts # => imprime nueva línea

nuevo_array = a.select { |num| num > 4 }
p nuevo_array # => [7, 8, 10]
nuevo_array = a.select { |num| num < 10 }
               .reject{ |num| num.even? }
p nuevo_array # => [1, 3, 7]

# Multiplicar cada elemento por 3 produciendo un nuevo array
nuevo_array = a.map {|x| x * 3}
p nuevo_array # => [3, 9, 12, 21, 24, 30]
```

Entonces...

- La API de arrays en ruby es muy flexible y poderosa.
- Existen muchas formas de procesar los elementos dentro de un array.